CCSLM Hardware Possibilities — Note to Self

Core observation

Concatenated Context Selected Linear Mappings (CCSLM) has a useful hardware property: its fundamental computation is a sequence of **linear mappings selected or modulated by context**.

$$y = L_n(C_n, x_n) \cdots L_2(C_2, x_2) L_1(C_1, x_1) x$$

This admits several implementations that do not require conventional neural-network hardware.

1. Direct analog implementation

Analog circuitry can implement the basic linear mapping

$$y = Wx$$

using relatively simple primitives:

- resistive or transconductance weighted sums;
- current-mode computation;
- current mirrors;
- capacitive/charge-domain computation;
- transistor conductance as a programmable weight;
- analog multiplexers and transmission gates for context selection.

The **data pathway** carries the analog vector, while a separate **context pathway** controls which mapping is active.

A context state could therefore select:

$$W_1,\ W_2,\ W_3, \ldots$$

without digitally calculating the matrix multiplication.

This is particularly attractive when the context has a small number of states.

2. Analog context modulation

Hard selection is not necessary.

A context signal could continuously control interpolation between mappings:

$$W(C) = \sum_k \alpha_k(C) W_k.$$

Current steering, variable transconductance, or programmable conductances could provide this naturally.

Thus CCSLM could range from discrete switched linear systems to continuously context-modulated analog computation.

3. Digital implementation

The same architecture maps naturally onto digital hardware.

A context decoder can select a region of parameter memory and feed its weights into a matrix/vector computation unit.

However, CCSLM does not necessarily require a conventional dense matrix multiplier. Its matrices can have considerable structure.

4. Fast Walsh-Hadamard Transform (FWHT)

The **FWHT is especially attractive as a hardware primitive** for CCSLM.

An N-dimensional Walsh-Hadamard transform requires only a sequence of additions and subtractions:

$$(a, b) \to (a + b,\ a - b).$$

Consequently, a large structured transformation can be constructed from many very small, highly regular operations.

This has several possible implementations:

- digital add/subtract networks;
- SIMD/vector hardware;
- bit-serial or low-precision arithmetic;
- analog summing/subtracting circuits;
- charge-domain circuits;
- transistor-level butterfly networks.

The regularity of the butterfly structure makes it particularly suitable for compact hardware.

5. Dense structured weight matrices

A useful CCSLM construction is therefore something like

$$W = HD_2HD_1H\cdots$$

where H represents a Walsh-Hadamard transform and D_i represents a simple diagonal operation, potentially containing signs or learned/scaled coefficients.

What appears to be a **dense matrix** can consequently be represented by a relatively small structured circuit.

This is important because a dense matrix need not imply a physically dense collection of independent multipliers.

The hardware can instead exploit the factorization:

$$\text{dense transformation} \quad \longrightarrow \quad \text{simple structured operations.}$$

6. Context-selected structured mappings

The two ideas combine naturally:

$$x \to H \to D(C) \to H \to D(C) \to H \to y.$$

Context can select the diagonal parameters, signs, or particular structured transformations at each stage.

This provides a potentially very efficient implementation of **context-dependent dense linear mappings**.

7. Mixed-signal CCSLM

A particularly interesting possibility is a mixed architecture:

- analog sensor signals provide context;
- simple comparators or state circuits discretize context;
- analog computation performs the linear mappings;
- FWHT-like networks provide structured mixing;
- digital memory stores or updates parameters;
- only selected intermediate quantities are digitized.

This could reduce the amount of data that must repeatedly cross an analog/digital boundary.

8. Hardware interpretation

CCSLM can therefore be regarded as a **context-addressed computational fabric** rather than necessarily as a conventional neural-network accelerator.

The important hardware resources are:

- **linear transformation elements;**
- **context-dependent switches/modulators;**
- **structured mixing networks;**
- **small parameter memories;**
- **cascaded computational stages.**

The separation between **context/control** and **data transformation** is particularly hardware-friendly.

Potentially important research direction

A useful question is not simply *"Can CCSLM run on analog hardware?"*

It is:

> **How much computational density can be obtained by combining context-selected analog linear mappings with highly structured transforms such as the FWHT?**

If the answer is favorable, CCSLM could potentially exploit hardware that is much simpler than a conventional architecture based on large numbers of independently programmable multipliers.